International Journal of Engineering and Applied Computer Science (IJEACS)

Importance of Testing in SDLC

Tanu Jindal

Department of Computer Science & Engineering
Noida Institute of Engineering & Technology (NIET)
Greater Noida, India.

***Abstract*— From previous year researches, it is concluded that testing is playing a vital role in the development of the software product. As, software testing is a single approach to assure the quality of the software so most of the development efforts are put on the software testing. But software testing is an expensive process and consumes a lot of time. So, testing should be start as early as possible in the development to control the money and time problems. Even, testing should be performed at every step in the software development life cycle (SDLC) which is a structured approach used in the development of the software product. Software testing is a tradeoff between budget, time and quality. Now a day, testing becomes a very important activity in terms of exposure, security, performance and usability. Hence, software testing faces a collection of challenges.**

Keywords — Software Engineering; SDLC; Software testing; Verification and Validation.

I. INTRODUCTION

Everyone knows the importance of computer in his life. In today's world computer is using in many fields like industry, education, transportation, medical, agriculture and research. Means, it is becoming an important element in the industry and advanced technology as well as developing countries. In today's life every field is dependent on computer for the betterment of their work. Also, with the help of computer a lot of time is saved. And time saving is indirectly a cost benefit approach. So software engineering is a best approach to develop a computer based system for every field in order to save the time and reduces its cost. Software Engineering is used to develop a system in a systematic way. For this purpose software/system development life cycle (SDLC) is used, as it is the process of developing the system with proper analysis, design, implementation and maintenance to improve the quality of the system. Even SDLC is a systematic approach for the development of the efficient system but without testing it is not possible. Because SDLC tells the process for the development of the system to improve the quality but doesn't helps in finding the defects of the system. So, testing plays an important role in software engineering.

II. RELATED WORK

Gelperin and Hetzel [4] presented the evolution of software test engineering which traced by examining changes in the testing process model and the level of professionalism over the years. Two phase models such as the demonstration and destruction models and two life cycle models such as the evolution and prevention models are given to describe the growth of software testing. Hamlet and taylor [10] presented more extensive simulations, and reach at more precise results about the relationship between partition probability, failure rate, and effectiveness. Vishwas Massey and K.J.Satao [7] in their paper have also compared various SDLC Models for performance and have also proposed a new model for better performance. But both the papers do not make a comparison between the research methodology and SDLC process. Richardson and Malley[1] proposed one of the earliest approaches focusing on utilizing specifications in selecting test cases. They proposed approaches to specification-based testing by extending a wide range of implementation-based testing techniques to be applicable to formal specification languages and determine these approaches for the Anna and Larch specification languages. Madeyski Lech et al.[9] presented the concept of using a set of second order mutants by applying them to large open source software with number of different algorithms. They show that second order mutation techniques can significantly improve the efficiency of mutation testing at a cost in the testing strength. Ntafos [2] presented the comparisons of random testing, partition testing and proportional partition testing. The author guaranteeing that partition testing has at least as high a probability of detecting a failure comes at the expense of decreasing its relative advantage over random testing. Juristo et al. [6] analyzed the maturity level of the knowledge about testing techniques. For this, they examined existing empirical studies about testing techniques. According to knowledge, they classified the testing techniques and choose parameters to compare them. J. A. Whittaker[3] presented a four phase approach to determine how bugs escape from testing. They offer testers to a group related problems that they can solve

ISBN: 978-0-9957075-1-1

during each phase. Claessen et al. [5] developed a lightweight and easy to use tool named "quickCheck", that is a combination of two old techniques (specifications as oracle and random testing) works extremely well for Haskell program. They present a number of case studies, in that the tool was successfully used and also point out some pitfalls to avoid. Harrold et al. [8] presented a new approach to class testing that supports data flow testing for data flow interaction in a class. They also describe class testing and the application of dataflow testing to class.

III. Functions And Goals Of Testing

The primary function of testing is to detect bugs. The scope of testing includes execution of that code in various environments and also to examine the aspects of the code - does the software do what it is supposed to do and function according to the specifications? Testing is an activity that is performed for evaluating software quality and also for improving it. The goal of testing is systematically and stepwise detection of different classes of errors within a minimum amount of time and also with a much less amount of effort. The basic purpose of testing is verification and validation in order to find various errors and problems – and the aim of finding those problems is to get them fixed. Testing is more than just error detection. Testing is done under controlled conditions.

Verification: To verify if system behaves as specified. It is the checking and testing of items, which includes software, for conformance and consistency of software by evaluating the results against pre-defined requirements. In verification we ask a question, are we building the product right?

Validation: In this we check the system correctness which is the process of checking that what has been specified by user and what the user actually wanted. In validation we ask a question: Are we building the right system?

IV. SDLC (Software Development Life Cycle)

SDLC serves as a guide to the project and provides a flexible and consistent medium to accommodate changes and perform the project to meet the client's objectives. There are various stages used in the life cycle of software development. Software development life cycle is basically a systematic way of developing software. It includes various phases starting from the functional requirement of software (means what software is supposed to do). After that designing takes place then development and then testing. After testing is finished, the source code is generally released for Unit Acceptance Testing (UAT) in client testing environment. After approval from client, the source code is released into production environment [11]. There is various software development approaches defined and designed which are used during development process of software, these approaches are also referred as "Software Development Process Models". Each process model follows a particular life cycle in order to ensure success in process of software development. SDLC phases define key schedule and delivery points which ensure timely and correct delivery to the client within budget and other constraints and project requirements. SDLC co-operates project control and management activities as they must be introduced within each phase of SDLC.

There is various software development approaches defined and designed which are used during development process of software, these approaches are also referred as "Software Development Process Models".

V. Phases In Software Development Lifecycle

Testing phase has much importance in SDLC due to a major role in debugging and error correction. The phases of SDLC is being followed in both testing and development cycle of any software application. Here are the phases of SDLC that is being followed:

a) *Requirements Gathering and Analysis:* Under this phase, proper requirements of project are gathered. All close functions are brought in to focus. All kinds of requirements and analysis of user requirement are done in this phase.

b) *System Design:* This is the next phase in SDLC where a rough system design is made. With all data and information being gathered, a system design is made.

c) *Development:* This is the next phase after system design when development of project is made. According to design, proper coding is done to gain that design. Programming language might be selected according to the project.

d) *System Testing:* Just after development phase, testing is carried out to know the outcome of application. Testing is made to know the actual result and the expected result.

e) *Operations and Maintenance:* This is the final stage of SDLC, where the software that is being developed is being distributed to end users who are responsible for maintaining and using it for proper operations. The software that is being developed must be open to any changes being made in coding.

VI. Role Of Testing In SDLC

Testing is required to remove the discrepancy in the software product development process. In order to implement any software product, it has passed through a set of various phases. With the help of testing we can catch small problems before they become big problems later on. Testing activities also provides the chance to review requirements for important quality attributes, to ask questions and to resolve issues earlier. There are many activities that are performed during the testing. These are:

- Test Analysis
- Test Design
- Test Execution

These activities are required to reduce the rework which results in reducing the cost and time. Software testing is an

ISBN: 978-0-9957075-1-1

ongoing process which we can't stop in between. Testing is required in SDLC due to the following reasons:

- To identify the errors
- To remove ambiguity
- To improve the reputation of the company
- To improve quality of the product
- To remove Hazards
- For verification and validation
- To improve reliability
- To improve cost
- To increase the usability

VII. CONCLUSION

From the above discussion it is clear that without testing it is not possible to implement an effective product. If the product is not effective then it will decrease the quality of the product. So, Testing is essential to improve the quality of the system as well as to the success of the overall effort. Testing is performed by developer end and customer end but it can ensure a performance of the product by predicting its behavior. In this paper, it is concluded that testing should be used in all the phases of SDLC and not in one or two phases. Life cycle of Software development is that type of structure which is imposed on the development process of the software product. As there are different activities involved in SDLC so, testing plays a different role in different-different phase. For timely readiness of the system testing is very important as it provides the visibility of the quality of the product at each step.

REFERENCES

[1] D. Richardson, O. O'Malley and C. Tittle, "Approaches to specification-based testing", ACM SIGSOFT Software Engineering Notes, Volume 14 , Issue 9, 1989, pp. 86 – 96.

[2] Ntafos Simeon C. "On comparisons of random, partition, and proportional partition testing." Software Engineering, IEEE Transactions on 27.10 (2001): 949-960.

[3] J. A. Whittaker, "What is Software Testing? And Why Is It So Hard?" IEEE Software, January 2000, pp. 70-79.

[4] D. Gelperin and B. Hetzel, "The Growth of Software Testing", Communications of the ACM, Volume 31 Issue 6, June 1988, pp. 687-695.

[5] Claessen Koen, and John Hughes. "QuickCheck: a lightweight tool for random testing of Haskell programs."

[6] Juristo Natalia, Ana M. Moreno, and Sira Vegas. "Reviewing 25 years of testing technique experiments." Empirical Software Engineering 9.1-2 (2004): 7-44.

[7] Glenford J. Myers, "The Art of Software Testing, Second Edition" Published by John Wiley & Sons, Inc., Hoboken, New Jersey.

[8] Harrold Mary Jean, and Gregg Rothermel. "Performing data flow testing on classes." ACM SIGSOFT Software Engineering Notes. Vol. 19. No. 5. ACM, 1994.

[9] Madeyski Lech et al. "Overcoming the Equivalent Mutant Problem: A Systematic Literature Review and a Comparative Experiment of Second Order Mutation." (2013): 1-1.[LR4]

[10] Hamlet Dick, and Ross Taylor. "Partition testing does not inspire confidence (program testing)." IEEE Transactions on Software Engineering 16.12 (1990): 1402-1411.

[11] Accessibility Summit. (2006). Public Sector NeedsBetter Guidance On Web Accessibility, E-Government Bulletin (Issue 226, 13 November 2006)http://www.ukoln.ac.uk/webfocus/events/meetings/accessibility-summit-2006-11/egovernment- 2006-11-13.php (Accessed August 30th2007)

AUTHOR PROFILE

Tanu Jindal is working as Assistant Professor in Computer Science & Engineering Department of Noida Institute of Engineering and Technology, Greater Noida. She received Master's degree in Software Engineering from Thapar University, Patiala. She is pursuing Ph.D. in Software Engineering. Her main research interests are Software Engineering and Grid Computing. She has guided many M.Tech. students for their thesis. She has guided many B.Tech students in their Projects. She has more than 10 publications in National and International Journals and Conferences.